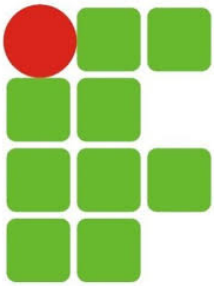


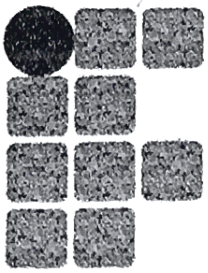
Instituto Federal de Educação Ciência
e Tecnologia do Rio Grande do Norte
Diretoria de Gestão de Tecnologia da Informação
Curso Médio Integrado em Informática

Relatório técnico da Construção de um Software para Análise de Expressão Gênica

Raquel Lopes de Oliveira

Orientador
Leonardo Ataide Minora, Mestre em Ciências da Computação
DIATINF/CNAT/IFRN

Natal(RN), fevereiro de 2016



Instituto Federal de Educação Ciência
e Tecnologia do Rio Grande do Norte
Diretoria de Gestão de Tecnologia da Informação
Curso Médio Integrado em Informática

Relatório técnico da Construção de um Software para Análise de Expressão Gênica

Raquel Lopes de Oliveira

Aprovado em: junho de 2016

Orientador
Leonardo Ataide Minora, Mestre em Ciências da Computação
DIATINF/CNAT/IFRN

Relatório técnico apresentado à DIATINF para a conclusão da Prática Profissional do Curso Técnico de Nível Médio Integrado em Informática, em cumprimento às exigências legais como requisito parcial à obtenção do título de Técnico em Informática.

Natal(RN), fevereiro de 2016

I wanna stand up, I wanna let go
You know, you know, no you don't, you don't
I wanna shine on, in the hearts of men
I want a meaning from the back of my broken hand
All These Things That I've Done, The Killers

Agradecimentos

Primeiramente gostaria de agradecer as pessoas mais importantes da minha vida, minha família. À meu pai, Ecildo, minha mãe, Joseana, e minha irmã, Eciana, por sempre acreditarem em mim, investirem em mim e apoiarem minhas escolhas.

Aos amigos da bolsa de manutenção, aos amigos de outros curso, amigos do curso de TADS, à turma *Inforever* que me acompanhou durante três dos quatro anos que vivênciei no IFRN, à todos os professores com que tive a oportunidade de aprender, à professora de geografia Maria Luiza que permitiu a única visita técnica da turma, ou seja, à todos que contribuíram nessa jornada.

Gostaria de citar Deyliane Souza, Lucas Torres e Regina Silva como colegas de classe que nunca deixaram de me ajudar e agradeço em especial à Sedir Morais que me ajudou e ajuda dentro e fora de minha vida acâdemica se mostrando uma pessoa cada vez mais especial em minha vida.

Sumário

Lista de Figuras

RESUMO

Criação de uma aplicação desktop Java para análise de expressão gênica e obtenção de uma assinatura gênica confiável e verificação de sua estabilidade a partir de dois arquivos de entradas referentes aos dados obtidos através de experimentos/estudos de microarray e/ou de sequenciamento de segunda geração.

Capítulo 1

Introdução

Esse trabalho teve início durande o curso de Tecnologia da Informação na UFRN quando os professores Jorge Estefano Santa de Souza, Silvia Maria Diniz Monteiro Maia e Carlos Eduardo da Silva propuseram a realização do mesmo em um trabalho das disciplinas de *LINGUAGEM DE PROGRAMAÇÃO II* e *ESTRUTURAS DE DADOS BÁSICAS II*. Como forma de otimizar tempo, e em sendo desenvolvimento de software, esse trabalho foi o suficiente para ser usado no trabalho de conclusão de curso para o Técnico em Informática no IFRN.

Esse projeto foi desenvolvido por mim, Raquel Lopes de Oliveira, e por Lucas Torres da Silva com participação de Jeffersson Galvão durante o ano de 2013. A implementação teve orientação da professora Silvia Maria Diniz Monteiro Maia e do professor Carlos Eduardo da Silva, ambos da UFRN; e o este relatório teve orientação do professor Leonardo Ataide Minora, IFRN. Este trabalho teve como o propósito principal criar um software para a análise de uma assinatura estável de expressão gênica.

"Uma assinatura de expressão gênica é obtida após comparar valores representativos de expressão gênicas entre grupos de amostras (por exemplo grupo de indivíduos sadios versus grupo de indivíduos doentes)." Nota de aula.

Para a solução do problema foi usada a linguagem Java.

1.1 Motivação

A principal motivação para o projeto foram suas aplicações, as pessoas que colaboraram para a realização do projeto, o feedback dado e a otimização do tempo entre as disciplinas da UFRN e o trabalho de conclusão de curso no IFRN.

1.2 Objetivo

1.2.1 Objetivo geral

Este trabalho teve como objetivo geral criar um software para a análise de expressão gênica.

1.2.2 Objetivo específico

"O objetivo deste projeto é identificar uma assinatura estável de expressão gênica capaz de predizer respostas a tratamentos através da análise combinada de dados de expressão gênica e dados clínicos."Nota de aula. Como objetivos específicos, foi contemplado no sofware desenvolvido um(a):

- estratégia para a eliminação de genes não informativos e obtenção da assinatura gênica;
- aplicação de um algoritmo de agrupamento;
- uso de clusterização;
- obtenção de uma matriz de distâncias;
- criação de um dendograma, em formato de árvores, dado a matriz de distância encontrada;
- checar a confiabilidade da assinatura atráves da validação cruzada.

1.3 Organização do trabalho

Este trabalho esta organizado com os seguintes capítulos: este capítulo 1 com uma introdução ao trabalho; o capítulo 2 com a descrição do desenvolvimento da implementação do software; e, por fim, o capítulo 3 com as considerações finais.

Capítulo 2

Desenvolvimento

2.1 Contextualização

O software de análise gênica é uma ferramenta de análise do gene em geral. Suas aplicações são variadas, sendo uma dessas a escolha mais adequada no tratamento de um paciente específico e/ou ter que identificar em qual grupo de epidemiologia o mesmo se encontra. Como se trata de uma análise geral também é possivel usar essa ferramenta para além da análise do gene humano como na análise em parasitas, fungos entre outros. Neste caso, uma possível aplicação seria num viveiro de camarão para a verificação se há contaminação, ou não.

2.2 Descrevendo a identificação de uma assinatura estável

Para verificar se a assinatura de uma expressão gênica é estável é necessário primeiramente gerar a assinatura gênica (seção 2.2.1) onde serão descartados os genes que não possuem significância estátistica a partir de um valor definido pelo usuário; uma análise é feita através do agrupamento (seção 2.2.3) onde os genes testados serão separados em dois (ou mais) grupos [depende do arquivo de entrada e se não há uma amostra classificada erroneamente]. Após a assinatura é validada através da validação cruzada

(seção 2.2.5) onde uma amostra é qualquer é escolhida e a análise de agrupamento(clausterização) é feita novamente, uma vez que a estrutura de mantém a amostra retorna ao dado original; esse processo se repete para todas as amostras. "Uma assinatura gênica é considerada estável se após as N configurações (remoções de amostras, uma a uma com reposição), a c lusterização inicial das amostras se mantém."Nota de aula.

2.2.1 Assinatura Gênica

Para obter a assinatura gênica primeiramente é necessário fazer o uso de um teste estatístico que tem como função os genes considerados significantes. O *p-valor* do teste é utilizado como critério de corte. Ou seja, o resultado da filtragem de genes é uma assinatura gênica, isto é, uma lista de genes significantes.

Neste software o usuário será capaz de escolher o teste a ser utilizado, porém com limitação, pois os testes disponíveis dependem da quantidade de grupos de estudo; também é possível escolher o *p-valor* de corte.

Para dois grupos, podem ser utilizados:

Teste U de Mann-Whitney Teste padrão do sistema. Testa se as expressões gênicas obtidas se originam de populações com a mesma distribuição.

Teste t de Student Testa se as expressões gênicas obtidas se originam de populações com expressão média distinta.

Teste de Kolmogorov-Smirnov Semelhante ao teste de Mann-Whitney.

Quando o estudo possue mais de dois grupos o único teste que pode ser utilizado é a Análise de Variância (ANOVA).

Mais informaçãoes sobre os testes estão na seção 2.4.4

2.2.2 Matriz de Distâncias

Após a filtragem é realizado uma matriz de distâncias. Seja m o número de genes na assinatura gênica e n o número de amostras, cada amostra de

índice i nós associamos um ponto m-dimensional $P_i = (E_{i,1}, E_{i,2}, \ldots, Ei, m)$.

A matriz de distância é uma matriz quadrada $M_{n,n}$, onde:

$$m_{i,j} = dist(P_i, P_j)$$

.

A distância entre dois pontos é determinada pela métrica, que pode ser escolhida pelo usuário (para verificar como veja a seção 2.6.3. Quatro métricas estão disponíveis no programa:

Euclidiana

$$dist_E(P_i, P_j) = \sqrt{\sum_{k=1}^{m}(E_{i,k} - E_{j,k})^2}$$

Euclidiana Quadrada

$$dist_{E^2}(P_i, P_j) = \sum_{k=1}^{m}(E_{i,k} - E_{j,k})^2$$

Manhattan

$$dist_M(P_i, P_j) = \sum_{k=1}^{m}|(E_{i,k} - E_{j,k})|$$

Máxima

$$dist_{max}(P_i, P_j) = max\{|(E_{i,k} - E_{j,k})|\}$$

É importante enfatizar que nenhuma normalização dos dados de expressão gênica é realizada pelo sistema, os dados vindos do arquivo de entrada já devem estar normalizados. Sendo assim, as métricas disponíveis só relacionam corretamente as amostras se uma normalização dos dados for feita previamente.

2.2.3 Agrupamento

Após a criação da escolha da matriz é necessário realizar o agrupamento hierárquico das amostras. O algoritmo responsável pelo agrupamento foi implementado na classe Clustering.java. Ele pega os 2 (dois) clusters mais

próximos a cada etapa e os combinada. A distância entre dois grupos é determinada por um critério de ligação. Foram implementados os critérios *Single Linkage*, *Complete Linkage* e *UPGMA* dentro do diretório */genesigno/clustering/linkage*. A escolha de qual critério utilizar é mostrada na seção 2.6.4.

2.2.4 Dendograma

Após realizar a clusterização, agrupamento, temos a definição de dendograma, exemplo: um gráfico que representa a árvore gerada após o agrupamento. Foi implementado um algoritmo capaz de transformar a representação hierárquica do agrupamento no dendograma. Como o programa foi implementado inteiramente com árvore binária, perfeito para 2 (dois) grupos. Logo quando se trata de 3 (três) ou mais grupos no estudo, o dendograma não apresentará uma árvore ternária, por exemplo no caso de 3 grupos. Para facilitar a comparação, grupos distintos são coloridos de forma distinta.

2.2.5 Validação Cruzada

Para validar a assinatura foi usada a validação cruzada: "O conceito central das técnicas de validação cruzada é o particionamento do conjunto de dados em subconjuntos mutualmente exclusivos, e posteriormente, utilizasse alguns destes subconjuntos para a estimação dos parâmetros do modelo e o restante dos subconjuntos são empregados na validação do modelo. Existem diversas formas de realizar o particionamento dos dados, sendo as três mais utilizadas: o método *holdout*, o *k-fold* e o *leave-one-out*"Nota de aula.

Foi implemetado a técnica *leave-one-out*. Usando esta técnica cada amostra é removida e a clausterização é feita novamente a fim de analisar se a mesma se mantém constante. Caso não se mantenha constante isso implica dizer que a assinatura gênica não é estável e portante a validação não obteve sucesso. Caso contrário a validação obteve sucesso e a assinatura gerada é uma assinatura estável.

Mesmo não implementando a generalização *k-fold* devido a sua complexidade, foi implementado uma generalização multigrupo onde é possível fazer

uma verificação, exemplo, uma validação para quando há mais de dois grupos.

Na seção 2.4.7 encontra-se mais informações sobre a validação cruzada.

2.3 Projeto Orientado a Objeto do Software

2.3.1 Tecnologias utilizadas

A linguagem de programação utilizada foi a Java com o compilador Oracle Java versão 8 update 73.Os seguintes softwares foram usados com as suas respectivas utilidades:

- *Java Development Kit 8u73*: conjunto de utilitários para compilar e executar os códigos-fontes;
- *Eclipse Luna*: para a edição dos códigos-fontes;
- *Dia 0.97.2*: para a realização do diagrama de classe UML.

2.3.2 Arquivos de entrada e saída

O software tem 2 (dois) arquivos de entrada e 1 (um) arquivo de saída, ao qual é descrito abaixo nesta sessão.

São 2 (dois) os arquivos de entrada, um contendo a distribuição das amostras em grupos e o outro indicando a expressão gênica encontrada para cada par de gene (amostra).

Cada linha do arquivo de grupos possui duas colunas identificando o nome da amostra e o nome do grupo correspondente. A imagem 2.1 mostra o arquivo exemplo com dois grupos: *caso* e *controle*, ele pode ser encontrado em: `https://gist.github.com/raquel-oliveira/f3e72b76da3d20a905b40ceb3fde5d30`. Não são aceitos estudos com um único grupo, porém é aceito com mais do que dois grupos.

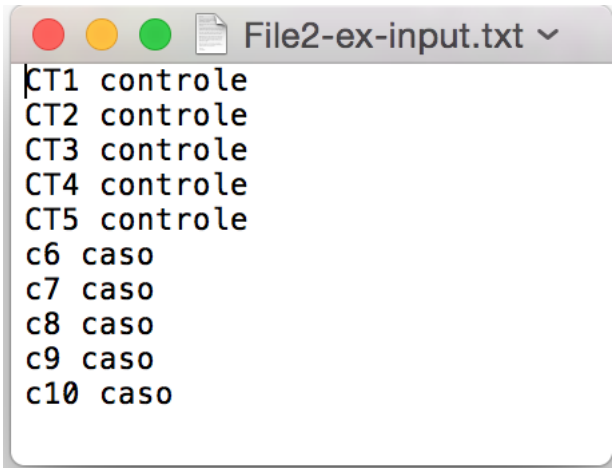


Figura 2.1: Exemplo de arquivo entrada: grupos

Para o segundo arquivo, onde encontra-se a expressão gênica para cada gene, existe um índice de uma dada amostra que corresponde a linha do arquivo na qual essa amostra está descrita. Cada linha do arquivo de expressão gênica contem $n + 1$ colunas, onde n é o número de amostras. A primeira coluna contém o ïomedo gene e as colunas restantes possuem os seus valores. Obrigatoriamente pontos flutuantes positivos correspondente a expressão do gene nas amostras, em ordem de indexação. Um exemplo de arquivo é apresentado na figura 2.1 e pode ser encontrado em `https://gist.github.com/raquel-oliveira/df364e2f53668d3f9acd78e5818aac0b`.

File1-ex-input.txt

Gene1	60.49	783.22	537.53	83.29	66.46	84.35	55.98	89.98	245.83	416.83
Gene2	39.62	35.81	24.63	12.03	25.86	248.77	227.84	355.61	593.57	140.81
Gene3	115.05	53.20	42.08	33.78	91.59	87.75	55.59	95.90	19.58	101.63
Gene4	206.29	148.45	461.41	328.67	264.74	474.42	540.51	285.45	82.30	231.59
Gene5	194.00	481.03	365.89	536.65	589.75	426.71	359.20	581.38	598.00	604.73
Gene6	221.12	290.70	81.09	466.50	104.26	13.12	259.95	280.23	78.41	92.22
Gene7	43.63	74.73	102.94	179.05	16.97	22.78	48.51	143.42	110.56	171.63
Gene8	10.61	2.69	2.54	6.34	14.10	2.70	5.49	14.03	0.43	6.92
Gene9	275.39	504.11	475.24	40.48	379.11	298.47	25.96	549.02	270.64	304.70
Gene10	36.27	280.71	439.49	426.72	369.80	419.06	173.86	364.51	429.54	10.23
Gene11	286.11	314.48	126.74	193.06	467.69	389.90	280.23	193.97	169.73	217.71
Gene12	333.84	611.94	198.25	547.44	576.22	387.51	218.90	119.21	341.63	587.38
Gene13	62.96	337.08	540.45	319.91	68.23	106.31	558.19	155.21	261.06	421.63
Gene14	262.67	246.06	7.91	14.76	89.08	574.35	659.63	615.62	465.50	161.34
Gene15	13.82	12.49	5.54	9.38	5.73	17.34	3.83	5.35	4.23	16.96
Gene16	136.60	879.30	291.39	212.07	618.49	689.32	207.40	733.06	323.30	741.09
Gene17	102.73	290.06	305.96	339.92	169.56	369.23	252.71	252.78	118.48	62.13
Gene18	7.76	45.07	51.26	13.07	5.67	20.57	15.29	46.37	20.85	9.79
Gene19	489.38	585.78	701.59	788.96	802.42	256.53	70.96	625.33	203.38	674.56
Gene20	455.09	31.61	597.12	455.32	409.37	567.43	203.03	39.72	374.48	501.84
Gene21	241.15	299.82	195.54	519.18	14.49	550.36	138.67	277.90	653.78	517.10
Gene22	30.16	6.15	578.69	525.83	644.93	140.04	96.37	539.22	543.62	106.92
Gene23	101.20	73.50	72.89	80.75	67.81	16.40	189.41	111.76	0.24	32.54
Gene24	717.89	4.76	900.27	974.33	243.23	737.03	653.00	234.01	935.19	367.72
Gene25	42.90	154.28	125.17	6.17	58.29	55.47	142.85	54.86	130.82	138.44
Gene26	499.07	292.90	411.15	589.64	508.81	157.14	162.42	290.13	658.49	376.51
Gene27	455.94	138.39	37.28	74.27	426.27	513.80	277.81	584.37	538.68	410.47
Gene28	211.72	257.01	301.40	504.08	455.92	598.78	259.33	478.93	756.52	114.07
Gene29	7.28	6.32	41.51	23.77	60.11	7.12	76.02	13.52	20.98	123.12
Gene30	277.50	56.19	7.26	32.00	37.89	199.14	18.29	224.24	221.07	228.68
Gene31	401.56	81.92	513.45	380.44	172.46	308.50	417.34	249.90	227.35	155.31
Gene32	295.17	201.92	632.47	494.02	606.02	330.72	47.95	90.04	456.29	223.06
Gene33	97.93	18.75	82.73	66.99	92.10	73.33	6.90	56.49	18.46	11.63
Gene34	105.21	54.57	128.61	103.27	63.15	69.17	1.25	121.17	132.05	22.35
Gene35	461.52	775.76	408.48	492.08	392.89	223.41	730.24	702.10	659.57	413.64
Gene36	187.72	120.07	13.28	269.53	121.74	580.42	621.43	354.70	269.82	209.57
Gene37	12.31	10.17	10.59	7.18	8.28	18.29	16.48	0.96	9.52	12.48
Gene38	9.27	7.92	2.06	21.10	10.57	7.70	13.71	9.09	19.43	5.14
Gene39	234.06	23.95	130.08	458.15	85.65	52.88	500.63	79.86	38.25	404.02
Gene40	100.95	101.61	124.89	97.55	168.24	78.57	160.23	67.31	51.39	104.19
Gene41	170.16	12.34	431.13	371.17	243.40	118.73	195.42	267.70	176.46	202.24
Gene42	20.99	139.72	238.13	99.85	159.33	173.65	206.66	150.63	14.55	94.99
Gene43	363.30	61.79	455.49	62.15	161.07	564.08	210.81	200.12	52.22	359.90
Gene44	148.78	343.08	308.54	246.21	149.54	121.55	35.45	14.31	449.26	427.80
Gene45	153.16	468.44	33.98	326.06	471.40	331.73	470.37	194.01	287.63	543.77
Gene46	392.48	399.36	5.17	91.01	242.58	114.24	13.66	182.66	66.33	33.73
Gene47	2.12	281.43	144.27	317.96	45.88	2.33	1.35	106.60	299.72	211.55
Gene48	38.24	114.09	45.32	65.89	126.22	44.42	28.92	13.78	111.23	118.08
Gene49	121.37	124.87	94.45	165.39	68.96	22.93	63.71	21.49	195.00	177.69
Gene50	132.94	83.42	377.92	327.51	177.75	206.31	43.01	308.08	22.67	368.43
Gene51	2.83	46.34	174.47	0.39	217.54	18.81	28.28	221.17	47.99	202.06
Gene52	25.24	168.13	50.14	17.32	27.83	45.57	120.88	47.94	7.53	107.86

Figura 2.2: Exemplo de arquivo de entrada: expressão gênica por gene

É possível verificar como os arquivos são inseridos na seção 2.6.1.

A partir dos dados lidos é montado um modelo do estudo, descrito na seção 2.3.3.

2.3.3 Modelo

Os diagramas de classe UML das figuras 2.3, 2.4, 2.5 apresentam a proposta e implementação das informações a serem processadas num paradigma orientado a objetos.

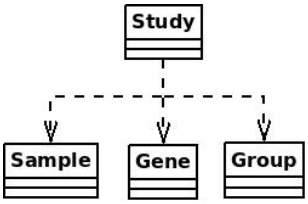


Figura 2.3: Arquitetura – Modelo

As informações obtidas nos arquivos são organizadas em um *modelo*, onde a entidade principal do modelo é o estudo (*Study*) que representa uma caso de teste, conforme mostra a figura 2.3. Cada estudo possui amostras (*Sample*), grupos(*Group*) e genes(*Gene*).

2.3.4 Assinatura Gênica

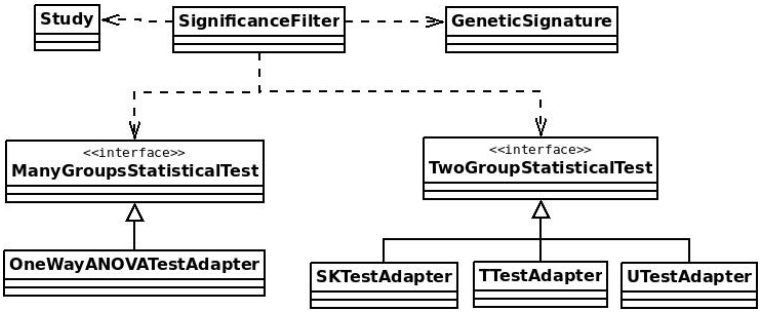


Figura 2.4: Arquitetura – Assinatura Gênica

As classes para assintatura gênica são exibidas no diagrama de classe UML na figura 2.4. A classe *SignificanceFilter* é responsável por filtrar os genes

de um estudo e gerar uma assinatura gênica (*GeneticSignature*). Seguindo o padrão de projeto *strategy*, cada teste estatístico disponível é uma class que herda de *TwoGroupStatisticalTest* e *ManyGroupsStatisticalTest*, para dois ou mais grupos respectivamente.

2.3.5 Clustering

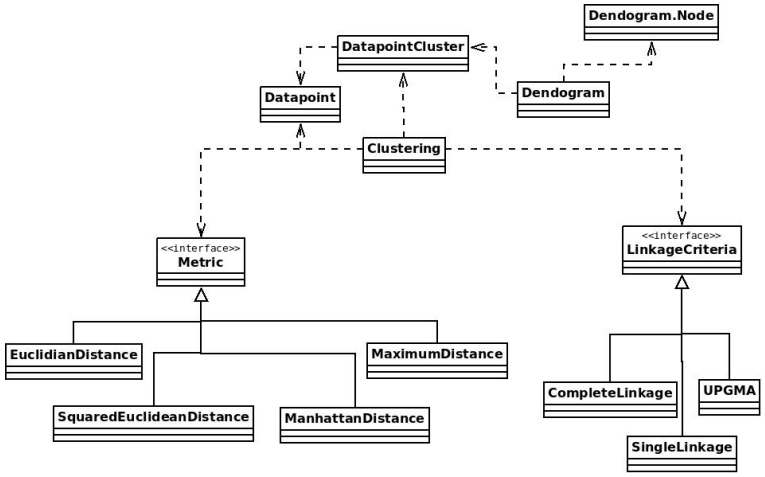


Figura 2.5: Arquitetura – Agrupamento

A classe *Clustering*, figura 2.5, coordena o processo de agrupamento. Seguindo o padrão *strategy*, essa classe pode receber um objeto *Metric* e um objeto *LinkageCriteria*. A classe *Datapoint* representa as coordenadas de uma amostra, e a classe *DatapointCluster* forma a floresta de *clusters*.

2.3.6 Validação Cruzada

O processo de validação é realizado pela classe *Validator*. Uma assinatura de validação é representada pela classe *ValidationSignature*.

2.3.7 Entrada e Saída

A classe java *InputFilesParser* é responsável por ler os arquivos de descrição e montar o estudo. Ela segue o padrão de projeto *builder* com classes específicas para montar os objetos conforme descrito nas seções anteriores.

Classes específicas são utilizadas para escrever cada arquivo de saída que utilizam a classe utilitária java *OutputFilesWriter*.

2.3.8 Interface Gráfica

As classes da interface gráfica seguem os padrões determinados pela biblioteca java Swing, e se comunicam com o Mediator para interagir com o resto do sistema. Sendo assim o padrão de projeto é semelhante ao *MVC*.

2.3.9 Integração

A integração entre o modelo com a interface gráfica, UI, foi realizado com o auxílio da classe *Mediator*. Abaixo temos uma figura referente as ligações da estrutura.

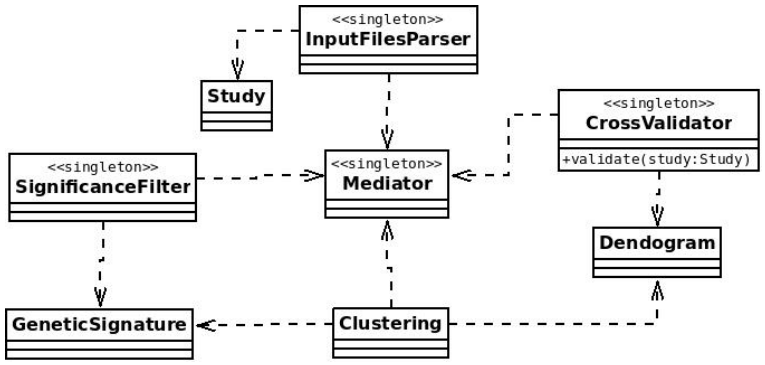


Figura 2.6: Arquitetura – Integração

> "A classe *Mediator* é um *singleton* responsável por coordenar os processos de leitura dos arquivos, determinação da assinatura gênica, geração de matriz de distância, clusterização, validação e escrita. Ela interliga os vários módulos, garantindo que as operações serão realizadas na ordem correta. Nesse sentido, ela segue o padrão de projeto *mediator*. A classe também oferece um ponto de acesso unificado para interação com a interface gráfica, seguindo o padrão de projeto *facade*.", Lucas Torres de Souza

2.4 Estruturas de dados utilizados

2.4.1 Floresta de clusters

"A floresta de clusters é uma floresta de árvores estritamente binárias. Essa estrutura se assemelha a estrutura de conjuntos dijuntos – ela possui os métodos *Create* e *Union* e a mesma restrição de que um nó pode estar somente em uma árvore. No entanto não há necessidade de um método *FindSet*, o que diminui a complexidade da estrutura. O método *Create(x)* criar uma nova árvore na floresta com um único nó x. O método *Union(x, y)* combina as árvores de raízes x e y. Ao contrario da estrutura tradicional de conjuntos dijuntos, um novo nó k é criado e utlizado como raíz da árvore combinada. [...]

A estrutura é implementada com um vetor de nós.[...] Cada nó pussui um filho esquerdo e um filho direito. Ao se criar um novo nó, o mesmo é adicionado ao vetor e marcado como raíz. Ao unir dois nós os mesmos são desmarcados como raiz e postos como filhos de um novo nó raiz. Para descobrir se um nó é raíz basta consutar o vetor auxiliar. Segundo essa implmentação, os métodos *Create(x)*, *Union(x, y)* e *IsRoot(x)* tem complexidade amortizada $O(1)$." Lucas T. de Souza.

2.4.2 Dendograma

Um dendograma é um gráfico de uma árvore estritamente binária como comentado na seção 2.2.4. Sua estrutura reflete a árvore de clusters resultante do processo de agrupamento, mas essa estrutura de dados foi implementada de forma a facilitar o desenho do dendograma. Tal árvore é construída de baixo para cima, utilizando uma visita em *pré-ordem* da árvore de clusters. Tal construção recursiva permite encontrar as características gráficas de cada nó, como coordenadas e cor.

2.4.3 Heap-Min

Um heap é uma estrutura em árvore utilizada para implementar uma fila de prioridades. Essa lista de prioridades não foi implementada por nós, ela faz parte da biblioteca padrão do Java. Em um heap-min o elemento de menor chave na fila pode ser encontrado em $O(1)$, removido em $O(\log n)$ e um novo elemento pode ser adicionado em $O(\log n)$. O heap foi utilizado para a análise de clusterização.

2.4.4 Assinatura Gênica

Para cada gene do estudo é necessário realizar um teste estatístico para encontrar o *p-valor* correspondente. Os testes estatísticos utilizados não foram implementados pelo grupo, e sua complexidade foi obitida na literatura. O teste t de Student, teste de Kolmogorov-Smirnov e a análise de variância tem complexidade linear. O teste U de Mann-Whitney tem complexidade $\Theta(n \log n)$.

Sendo assim, o processo de determinação da análise gênica tem complexidade $O(mn \log n)$.

Vale lembrar que apenas genes cujos testes determinaram um p-valor menor ou igual ao ponto de corte determinado pelo usuário são adicionados a assinatura gênica.

2.4.5 Cálculo da matriz de distâncias

Nessa etapa o software recebe uma assinatura gênica e produz uma matriz de distânicas entre amostras. A complexidade total do processo é $\Theta(n^2 m_g)$, pois segundo as métricas estabelecidas, o cálculo da distância entre duas amostras possui complexidade $\Theta(m_g)$.

2.4.6 Agrupamento

Nessa etapa o software recebe uma matriz de distâncias e gera uma hierarquia de grupos (*clusters*) de amostras. Trata-se de um algoritmo associativo pois ele se inicia com vários grupos unitários e então os combina até formar

um grupo total. Ele é hierárquico pois o seu resultado não é apenas um conjunto de clusters, mas uma árvore de clusters. O algoritmo implementado agrupa os clusters de dois em dois. Como resultado, a árvore de clusters gerada é uma árvore estritamente binária como comentado na seção 2.2.4 o algoritmo não pode gerar um dendograma de árvores diferentes de binárias. Assim, a árvore gerada possuí n folhas e $n-1$ nós internos. A distância entre dois clusters é determinada por um critério de ligação, como apresentado na seção 2.3.5. A cada etapa do algoritmo os dois clusters mais próximos segundo o critério de ligação são combinados. Para encontrar o próximo par a ser combinado, utilizamos uma heap-min na qual as chaves são as distâncias e os valores são pares de cluster. Um vetor adicional é utilizado para indicar quais clusters são raízes, isto é, ainda não foram agrupados. A estrutura em árvore utilizada para o processamneto está descrita na seção 2.4.1.

Encontra-se que a complexidade do algoritmo de agrupamento é $O(n^2 \log n)$. A solução encontrada é semelhante a do algoritmo de *Kruskal*. Segue abaixo o pseudo-algoritmo escrito por Lucas para melhor mostrar entendimento.

[H]

H : HeapMin, BF : BinaryForest i = 1 $\rightarrow$ n BF.create(i) i = j $\rightarrow$ n H.add(dist[i][j], (i, j)); — $O(n^2)$

H is not empty d, (i, j) = H.pop() — $O(n^2 \log n)$
BF.IsRoot(i) and BF.IsRoot(j) $k \leftarrow$ BF.Union(i, j) — $O(n)$
each root r distance$[r][k]$ = combine(distance$[r][i]$, distance$[r][j]$) — $O(n^2)$
H.add(distance$[r][k]$, (r, k)) — $O(n^2 \log n)$
Agrupamento Hierárquico

2.4.7 Validação Cruzada

A validação cruzada *leave-one-out* implementada consiste em repetir o processo de clusterização retirando cada amostra uma a uma e verificando a consistencia da nova árvore de clusters com a árvore originalmente gerada. Uma árvore de clusters é consistente com a árvore original se todos os pares de amostras que se encontram em lados opostos da nova árvore também estão em lados opostos na árvore original.

"Para verificar a validade de uma árvore, primeiro uma arranjo contendo o lado (esquerda ou direita) de cada amostra na árvore, utilizando o índice da amostra ($O(n)$). Em seguida, o lado de cada folha da nova árvore é comparada com os lados correspondentes no arranjo. A árvore é válida se todos as comparações obtiverem o mesmo resultado (lado igual ou diferente). O agrupamento precisa ser repetido n vezes nessa etapa, mas a matriz de distância pode ser reutilizada. Como resultado, a validação *leave-one-out* completa possuí complexidade $O(n^3 \log n)$." Lucas T. de Souza

2.5 Como compilar e executar

2.5.1 Como Compilar

2.5.2 Gerar executável

Usando o *Eclipse Luna* clique em ***File*** no menu superior, depois em ***Export*** e finalmente em ***Runnable JAR File***.

2.5.3 Como Executar

Em ambiente gráfico do usuário, clicar 2 (duas) vezes no arquivo executável jar *genesigno.jar*, já criado conforme a seção 2.5.2.

Pelo terminal, entre no diretório do executável,ß inserir o comando: $

```
java -jar genesigno.jar
```

2.6 Descrevendo a interface gráfica do usuário

Nesta seção será apresentado as interfaces gráficas do usuário bem como instruções de como utilizá-las. A tela inicial do software é apresentada na figura 2.7.

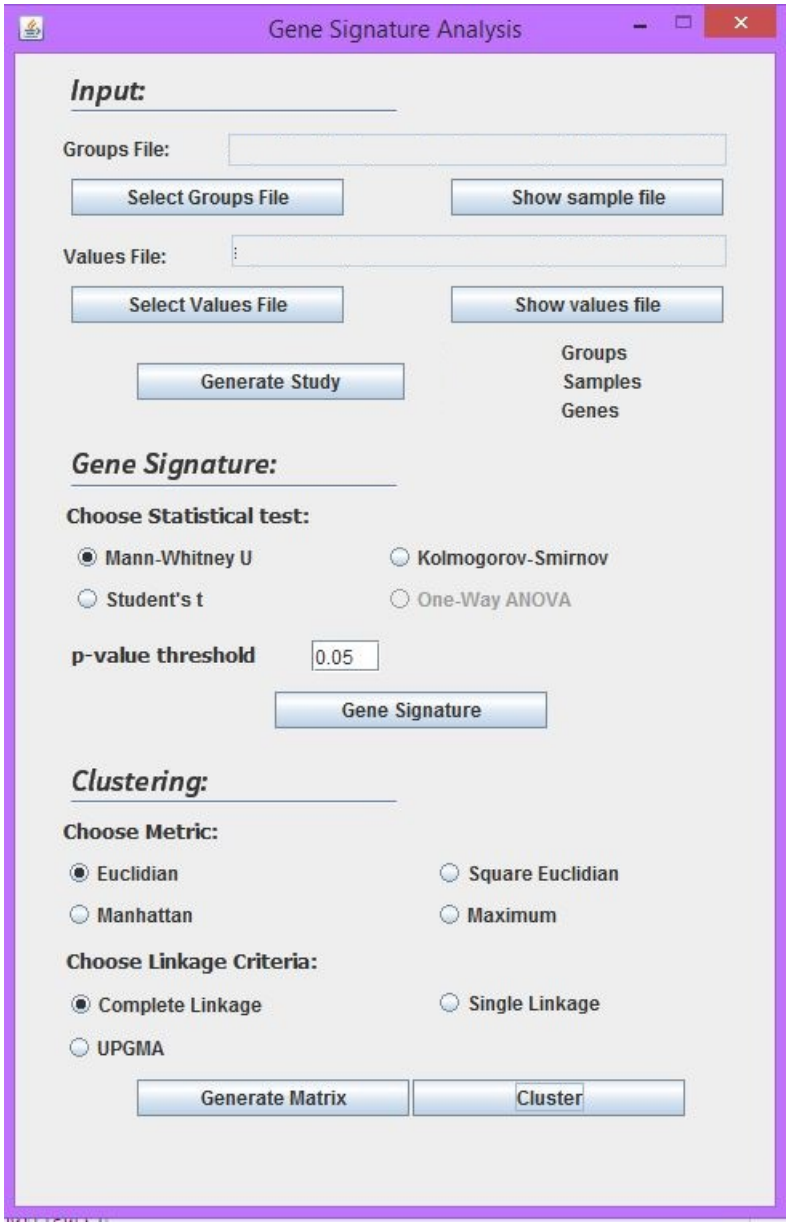


Figura 2.7: Tela inicial do software

2.6.1 Informando os arquivos de entrada

O primeiro passo ao executar o software é inserir os 2 (dois) arquivos com os dados de expressão normalizados.

Para tanto, na tela inicial, figura 2.7, clique em *Select Sample File* e selecione o arquivo que informa os grupos correspondentes as amostras; ao selecionar um arquivo o botão é habilitado *Show sample file* para visualização e/ou verificação do conteúdo do arquivo.

Ainda na tela principal, figura 2.7, clique em *Select Values File* para selecionar o arquivo que informa a expressão dos genes de cada amostra; após selecionar, o conteúdo do arquivo pode ser visualizado no campo botão *Show values file*.

Após os dois arquivos serem selecionado o botão *Generate Study* é habilitado. Clicando nele o *estudo* será gerado. Após o *estudo* ser gerado, será habilitado alguns itens na interface gráfica que permitiram configurar a geração da assinatura genica como descrito na subseção seguinte.

2.6.2 Configurando a geração da assinatura genica

De acordo com a quantidade de grupos existentes será possível escolher o teste estatístico. Caso possua mais de 2 (dois) grupos o único o únicoo teste disponível será o *Test One*. Caso seja 2 (dois) grupos será possível escolher entre o *Student t*, o *Mann-Whitney U* e o *Kolmogorov-Smirnov*.

É possível modificar o $p - valor$ para o critério de corte alterando o valor na caixa de texto ao lado de *p-value threshold*. Após definir a configuração, para gerar a assinatura genica sem os genes não informativos, basta clicar no botão *Gene Signature*.

Quando a assinatura genica é gerada, será apresentado outra tela com o resultado, sendo possível salvar esses valores através do botão *Save Gene Signature*, conforme apresentado na figura 2.8.

Para continuar o processo você pode fechar a tela de assinatura genica, ou não, e será possível gerar uma matriz de distâncias, conforme apresenta a próxima subseção.

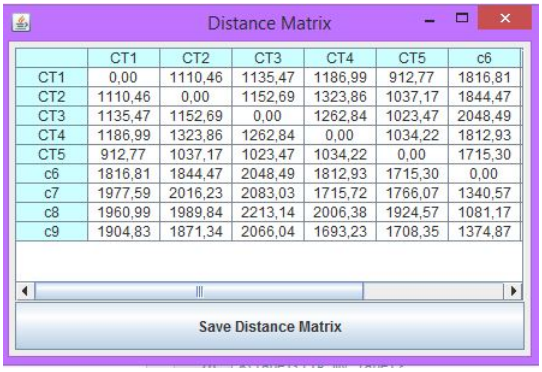
Figura 2.8: Tela com a simulação de uma assinatura gência

2.6.3 Gerando a matriz de distâncias

Os métodos para gerar a matriz de distâncias pode ser escolhida entre o *Euclidian*, o *Square Euclidian*, o *Manhattan* e o *Maximum*, conforme apresentado na figura 2.7. A escolha é habilitada após a geração de assinaturas genicas. Após escolher o método será possível gerar uma matriz de distâncias clicando no botão *Generate Matrix*.

Ao gerar a matrix de distâncias uma nova tela será exibida, ao qual mostra a matriz de distância gerada, permitindo salvar esta através do botão *Save Distance Matrix*, conforme figura 2.9.

Após a geração da matriz de distância, será possível fazer o agrupamento para finalmente ser visualizado o dendograma, conforme descreve a sub-seção a seguir.



	CT1	CT2	CT3	CT4	CT5	c6
CT1	0,00	1110,46	1135,47	1186,99	912,77	1816,81
CT2	1110,46	0,00	1152,69	1323,86	1037,17	1844,47
CT3	1135,47	1152,69	0,00	1262,84	1023,47	2048,49
CT4	1186,99	1323,86	1262,84	0,00	1034,22	1812,93
CT5	912,77	1037,17	1023,47	1034,22	0,00	1715,30
c6	1816,81	1844,47	2048,49	1812,93	1715,30	0,00
c7	1977,59	2016,23	2083,03	1715,72	1766,07	1340,57
c8	1960,99	1989,84	2213,14	2006,38	1924,57	1081,17
c9	1904,83	1871,34	2066,04	1693,23	1708,35	1374,87

Figura 2.9: Simulação da exibição da matriz de distâncias

2.6.4 Escolha do critério de ligação

O critério de ligação padrão é o *Complete Linkage* mas o software ainda permite selecionar outros 2 (dois), a saber: *Single Linkage* e *UPGMA*. Após selecionar o critério, pode-se clicar no botão *Cluster* para gerar o agrupamento, conforme mostra a figura 2.10.

Quando o agrupamento for realizado, uma nova tela será exibida com o dendograma gerado. Esta tela tem opções para: salvar a árvore que representa o dendograma, botão *Save tree*; salvar a imagem do dendograma, botão *Save Dendogram*; e validação da assinatura de expressão gência através do botão *Validate*. A figura 2.11 apresenta essa nova tela.

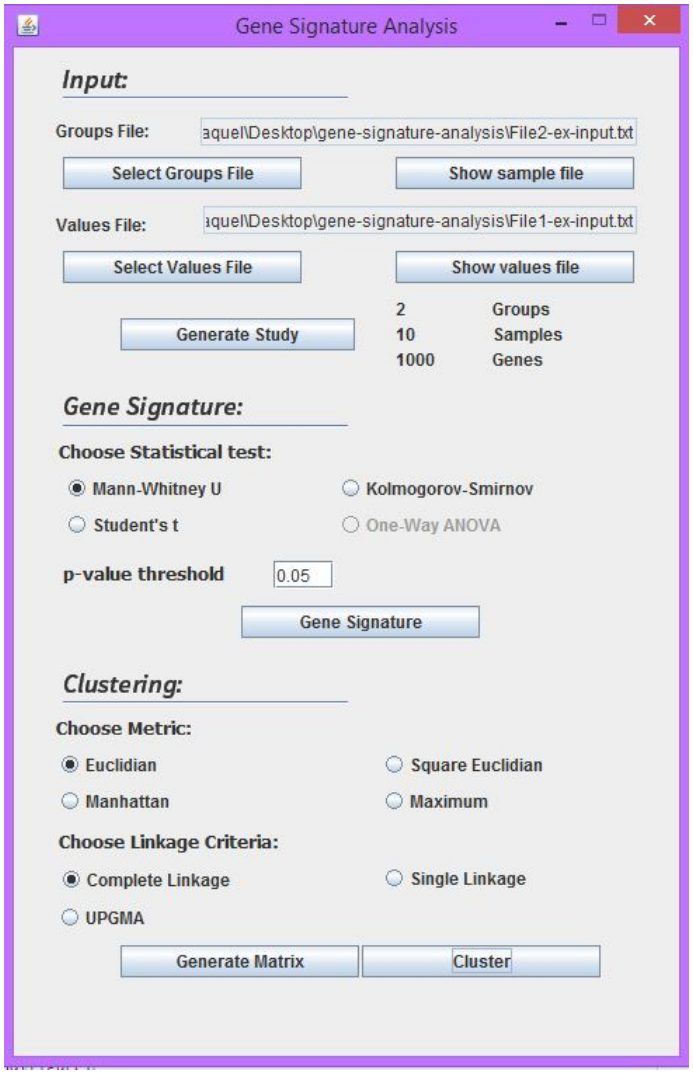


Figura 2.10: Tela inicial com a escolha do critério de ligação

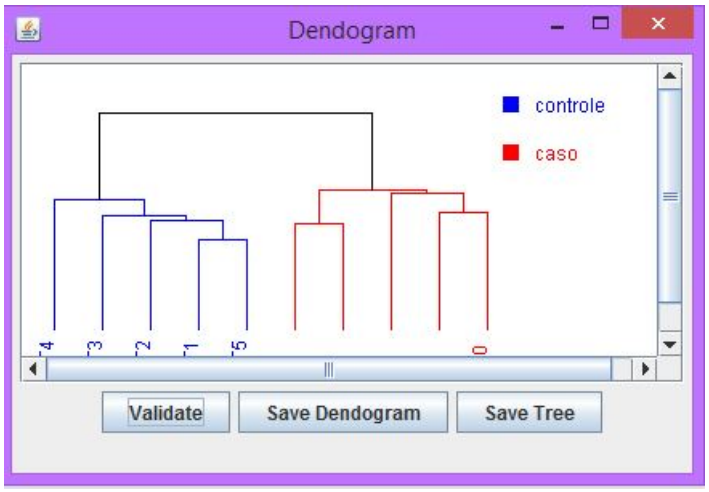


Figura 2.11: Tela para apresentar o dendograma

Capítulo 3

Considerações finais

O software para a análise de expressão genica foi construído e seu código-fonte encontra-se disponível em `https://github.com/raquel-oliveira/gene-signature`. Todos os objetivos específicos foram atendidos nesta implementação.

Ainda, este trabalho de conclusão de curso me permitiu algumas constatações, à saber:

- Os conhecimentos do IFRN e da UFRN foram complementares para a realização desse trabalho;
- Os conhecimentos adquiridos no IFRN permitiram a possibilidade de ser aprovada nas disciplinas na UFRN;
- É possível realizar projetos reais e não somente projetos unicamente para praticar os conteúdos aprendidos em disciplinas;
- É possível criar um software útil para uma determinada área/assunto sem ter necessariamente um conhecimento apronfudado sobre o mesmo.